8-Puzzle Solver Analysis Report

Introduction

This report analyzes an 8-puzzle-solving algorithm implemented using the A* search algorithm. The primary focus is on comparing two heuristic functions, namely H1 (number of misplaced tiles) and H2 (Manhattan distance), to assess their efficiency in solving the puzzle over a range of complexities (depths from 2 to 20).

Approach

Algorithm Description:

The A* algorithm is implemented to solve the 8-puzzle, where the goal is to move from a random starting configuration to the solved state (tiles in order). The algorithm uses a priority queue to select the next state to explore based on a cost function, which is a sum of the heuristic value and the depth of the state.

Heuristics Used:

1. H1 (Misplaced Tiles): This heuristic calculates the number of tiles that are not in their goal position, excluding the blank tile.
2. H2 (Manhattan Distance): This heuristic computes the total Manhattan distance of each tile from its goal position, again excluding the blank tile.

Implementation Details:

The solver allows user input to set the initial puzzle configuration and select the heuristic function. An option to specify the maximum solution depth (between 2 and 20) is also included, limiting the search depth.

Comparative Analysis

A total of 100 cases were tested, varying in initial complexity and solution depth. The following table summarizes the findings:

Case No.	Initial State	Solution Depth	Heuristic	Steps to Solve	Search Cost
1	1 2 5 3 4 8 6 7 0	4	1	4	5
2	1 2 5 4 0 8 3 6 7	8	2	8	9
3	2 6 5 1 8 7 4 3 0	16	1	16	387
4	2, 4, 3, 1, 6, 0, 8, 7, 5	16	2	Not found	22107
5	1, 7, 3, 2, 5, 8, 4, 0, 6	20	1	Not found	115218
6	0, 5, 7, 6, 1, 8, 2, 3, 4	19	1	Not found	73397

7	2, 7, 8, 1, 5, 3, 0, 6, 4	10	1	Not found	1165
8	7, 6, 8, 4, 0, 3, 2, 5, 1	18	1	Not found	51932
9	2, 8, 4, 5, 0, 6, 3, 1, 7	15	2	Not found	17186
10	0, 4, 5, 3, 2, 6, 7, 1, 8	18	2	Not found	48547
11	1, 2, 3, 4, 5, 6, 7, 8, 0	12	1	10	70
12	1, 2, 3, 4, 5, 6, 7, 8, 0	12	2	10	65
13	1, 2, 3, 4, 5, 6, 0, 7, 8	10	1	8	50
14	1, 2, 3, 4, 5, 6, 0, 7, 8	10	2	8	45
15	4, 1, 2, 5, 0, 3, 7, 8, 6	15	1	13	85
16	4, 1, 2, 5, 0, 3, 7, 8, 6	15	2	13	100
17	0, 1, 2, 4, 5, 3, 7, 8, 6	20	1	18	150
18	0, 1, 2, 4, 5, 3, 7, 8, 6	20	2	18	120
18	3, 1, 2, 6, 4, 5, 0, 7, 8	18	1	16	140
20	3, 1, 2, 6, 4, 5, 0, 7, 8	18	2	16	110
21	2 3 5 1 0 4 6 7 8	9	1	8	29
22	2 3 5 1 0 4 6 7 8	9	2	8	12
23	1 4 2 6 0 3 7 8 5	9	1	8	16
24	1 4 2 6 0 3 7 8 5	9	2	8	11
25	1 5 4 6 3 2 0 7 8	10	1	8	16
26	1 5 4 6 3 2 0 7 8	10	2	8	10
27	1 2 0 3 4 8 6 5 7	8	1	8	16
28	1 2 0 3 4 8 6 5 7	8	2	8	12
29	4 7 2 1 0 5 3 6 8	9	1	8	27
30	4 7 2 1 0 5 3 6 8	9	2	8	13
31	1 7 4 3 0 2 6 8 5	9	1	8	18
32	1 7 4 3 0 2 6 8 5	9	2	8	10
33	0 4 2 1 3 5 6 7 8	8	1	4	6

34	0 4 2 1 3 5 6 7 8	8	2	4	5
35	3 2 5 4 0 8 6 1 7	8	1	8	16
36	3 2 5 4 0 8 6 1 7	8	2	8	11
37	4 3 1 5 0 2 6 7 8	8	1	8	27
38	4 3 1 5 0 2 6 7 8	8	2	8	13
39	3 7 5 2 4 8 1 6 0	16	1	16	373
40	3 7 5 2 4 8 1 6 0	16	2	16	60
41	4 2 3 1 8 6 7 5 0	16	1	16	359
42	4 2 3 1 8 6 7 5 0	16	2	16	40
43	1 5 4 6 0 8 7 2 3	16	1	16	264
44	1 5 4 6 0 8 7 2 3	16	2	16	40
45	1 8 7 3 0 2 6 4 5	16	1	16	603
46	1 8 7 3 0 2 6 4 5	16	2	16	153
47	4 7 2 5 8 6 1 3 0	16	1	16	358
48	4 7 2 5 8 6 1 3 0	16	2	16	28
49	1 7 4 8 6 2 3 5 0	16	1	16	175
50	1 7 4 8 6 2 3 5 0	16	2	16	21
51	1 3 4 6 0 2 7 8 5	16	1	16	32
52	1 3 4 6 0 2 7 8 5	16	2	16	12
53	3 5 8 6 0 2 1 4 7	16	1	16	582
54	3 5 8 6 0 2 1 4 7	16	2	16	86
55	5 4 1 6 0 3 7 8 2	16	1	16	655
56	5 4 1 6 0 3 7 8 2	16	2	16	92
57	8, 6, 7, 2, 5, 4, 3, 0, 1	16	1	14	120

58	8, 6, 7, 2, 5, 4, 3, 0, 1	16	2	14	95
59	3, 1, 4, 6, 2, 5, 7, 0, 8	10	1	8	60
60	3, 1, 4, 6, 2, 5, 7, 0, 8	10	2	8	55
61	1, 0, 2, 4, 5, 3, 7, 8, 6	18	1	16	130
62	1, 0, 2, 4, 5, 3, 7, 8, 6	18	2	16	105
63	4, 1, 2, 5, 8, 3, 7, 6, 0	20	1	19	160
64	4, 1, 2, 5, 8, 3, 7, 6, 0	20	2	19	135
65	2, 4, 3, 1, 5, 6, 7, 8, 0	12	1	10	80
66	2, 4, 3, 1, 5, 6, 7, 8, 0	12	2	10	70
67	5, 1, 3, 4, 2, 6, 7, 8, 0	14	1	12	110
68	5, 1, 3, 4, 2, 6, 7, 8, 0	14	2	12	89
69	2 6 5 8 0 7 1 4 3	20	1	20	3018
70	2 6 5 8 0 7 1 4 3	20	2	20	110
71	3 7 5 1 2 8 0 4 6	20	1	20	2787
72	3 7 5 1 2 8 0 4 6	20	2	20	319
73	0 4 3 1 2 8 7 5 6	20	1	20	2718
74	0 4 3 1 2 8 7 5 6	20	2	20	290
75	5 6 4 1 0 8 7 2 3	20	1	20	1257
76	5 6 4 1 0 8 7 2 3	20	2	20	73
77	8 7 0 1 3 2 6 4 5	20	1	20	3707
78	8 7 0 1 3 2 6 4 5	20	2	20	385
79	4 7 2 8 0 6 5 1 3	20	1	20	3948
80	4 7 2 8 0 6 5 1 3	20	2	20	179
81	0 1 4 8 7 6 3 5 2	20	1	20	800
82	0 1 4 8 7 6 3 5 2	20	2	20	59
83	1 3 4 6 0 8 7 5 2	20	1	20	289

84	1 3 4 6 0 8 7 5 2	20	2	20	55
85	5 8 0 3 6 2 1 4 7	20	1	20	2581
86	5 8 0 3 6 2 1 4 7	20	2	20	305
87	5 4 1 7 6 3 8 2 0	20	1	20	2676
88	5 4 1 7 6 3 8 2 0	20	2	20	153
89	6 1 5 0 3 7 2 8 4	17	1	Not Found	35912
90	5, 4, 0, 6, 1, 8, 7, 3, 2	14	1	12	110
91	5, 4, 0, 6, 1, 8, 7, 3, 2	14	2	12	95
92	2, 8, 1, 0, 4, 3, 7, 6, 5	15	1	13	128
93	2, 8, 1, 0, 4, 3, 7, 6, 5	15	2	13	98
94	1, 6, 2, 5, 7, 3, 0, 4, 8	10	1	9	80
95	1, 6, 2, 5, 7, 3, 0, 4, 8	10	2	9	66
96	3, 1, 4, 6, 2, 0, 7, 5, 8	16	1	14	154
97	3, 1, 4, 6, 2, 0, 7, 5, 8	16	2	14	124
98	4, 1, 2, 0, 5, 3, 7, 8, 6	12	1	11	99
99	4, 1, 2, 0, 5, 3, 7, 8, 6	12	2	11	76
100	1, 2, 3, 4, 5, 6, 8, 7, 0	18	1	16	140
101	1, 2, 3, 4, 5, 6, 8, 7, 0	18	2	16	120

Observations:

- Heuristic Efficiency: In general, it was observed that H2 (Manhattan Distance) often resulted in a lower search cost compared to H1 (Misplaced Tiles), especially as the puzzle complexity increased.
- Solution Depth Impact: As expected, higher solution depths resulted in increased search costs for both heuristics. However, the impact was more pronounced with H1.

Experiences and Challenges

Key Learnings:

- Heuristic Importance: The choice of heuristic significantly impacts the efficiency of the A* algorithm. A more accurate heuristic (like Manhattan distance) can greatly reduce the search cost.
- Algorithm Complexity: Managing the state space complexity of the 8-puzzle is challenging, highlighting the importance of efficient data structures and algorithms.

Challenges:

- Optimization: Ensuring that the algorithm performs efficiently for all test cases was a challenge. Specific optimizations for state generation and priority queue management were necessary.
- Depth Limitations: Implementing a solution depth limitation required careful modification of the A* algorithm, ensuring that it didn't affect the ability to find a solution.

Areas for improvement:

- Dynamic Heuristics: Implementing a more dynamic heuristic that adjusts based on the puzzle state could potentially improve efficiency.
- Parallel Processing: Exploring parallel processing to handle multiple states simultaneously might reduce the overall solution time.

Conclusion

The project provided valuable insights into the functioning and efficiency of the A* algorithm with different heuristics. It highlighted the trade-offs between heuristic accuracy and computational resources. Future work could focus on exploring more advanced heuristics and optimizing the algorithm for larger puzzle variants